UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/728,441	12/01/2000	Wen-mei W. Hwu	042302 0269900	3226

27498 7590 10/19/2004

PILLSBURY WINTHROP LLP
2475 HANOVER STREET
PALO ALTO, CA 94304-1114

EXAMINER	
LI, AIMEE J	

ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 10/19/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/728,441	HWU ET AL.	
	Examiner	Art Unit	
	Aimee J Li	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 April 2004 and 14 July 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-52* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-52* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *30 June 2004*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-50 and new claims 51-52 have been considered. Claims 8, 10, 12, 14, 19-21, 32, 35-36, 38, 42-44, and 50 have been amended as per Applicant's request. New claims 51-52 have been added as per Applicant's request.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment filed 05 April 2004, IDS filed 30 June 2004, and Amendment filed 14 July 2004.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. Claims 1-3 are rejected under 35 U.S.C. 102(e) as being taught by Fleck et al., U.S. Patent Number 6,085,315 (herein referred to as Fleck).

5. Referring to claim 1, Fleck has taught a processor comprising:

   a. A functional unit adapted to execute an instruction issued to it from a dispatch stage (Fleck column 1, line 30 to column 2, line 2; column 2, lines 26-44; and Figure 1); and

b. A buffer in the dispatch stage coupled to the functional unit adapted to store a plurality of the instructions before issue to the functional unit (Fleck column 1, line 30 to column 2, line 2; column 2, lines 45-67; and Figure 1).

6. Referring to claim 2, Fleck has taught a processor according to claim 1, further comprising a decode stage register coupled between the dispatch stage and the functional unit, the functional unit coupled to the decode stage register for executing the instruction issued to the decode stage register from the dispatch stage (Fleck column 2, lines 35-55; column 3, lines 19-31; and Figure 1).

7. Referring to claim 3, Fleck has taught a processor according to claim 1, further comprising control logic coupled to the buffer adapted to cause a certain one of the stored plurality o instructions to be issued to the functional unit in accordance with a loop iteration stage (Fleck column 2, lines 45-67; column 4, lines 9-56; Figure 1; and Figure 3).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 4-18, 26-33, 35-41, and 43-49 rejected under 35 U.S.C. 103(a) as being unpatentable over Fleck et al., U.S. Patent Number 6,085,315 (herein referred to as Fleck) in view of Subramanian et al., U.S. Patent Number 5,867,711 (herein referred to as Subramanian).

10. Referring to claims 4-14, Fleck has taught

a. Control logic coupled to the buffer (Applicant's claim 8) (Fleck column 2, lines 45-67; column 4, lines 9-56; Figure 1; and Figure 3);

b. The control logic including:

i. A loop iteration register for storing a loop iteration parameter (Applicant's claims 8, 10, 12, and 14) (Fleck column 3, line 48 to column 4, line 56 and Figure 3), and

ii. A loop cycles register for storing a loop cycles parameter (Applicant's claims 8, 10, 12, and 14) (Fleck column 3, line 48 to column 4, line 56 and Figure 3);

c. Wherein the control logic is adapted to cause the plurality of instructions to be issued to the functional unit from the buffer in accordance with the loop iteration parameter and the loop cycles parameter (Applicant's claim 8) (Fleck column 3, line 48 to column 4, line 56 and Figure 3); and

d. Control logic coupled to the buffer, the control logic being operative so that the functional unit executes a number of loop iterations (Applicant's claims 9, 11, and 13) (Fleck column 1, line 55 to column 2, line 2; column 3, line 48 to column 4, line 56; Figure 1; and Figure 3).

11. Fleck has not taught:

a. Wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with a cycle within the loop iteration stage (Applicant's claim 4);

b. Wherein the buffer is further adapted to store a plurality of loop stage bit masks respectively associated with the plurality of instructions (Applicant's claim 5);

c. Wherein the buffer is further adapted to store a plurality of loop stage bit masks respectively associated with the plurality of instructions (Applicant's claims 6 and 7);

d. Wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with the respective one of the loop stage bit masks associated with the certain one of the instructions (Applicant's claims 6 and 7);

e. The control logic including an initiation interval register for storing a loop iteration initiation parameter (Applicant's claims 8, 10, 12 and 14); and

f. Wherein the stored plurality of instructions comprises a kernel set of loop instructions, a prologue set of loop instructions different than the kernel set of loop instructions, and an epilogue set of loop instructions different than the kernel set of loop instructions in accordance with the kernel set of loop instructions and received loop parameters (Applicant's claims 9, 11, and 13).

12. Subramanian has taught:

a. Wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with a cycle within the loop iteration stage (Applicant's claim 4) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figures 4; and Figure 5).

b. Wherein the buffer is further adapted to store a plurality of loop stage bit masks respectively associated with the plurality of instructions (Applicant's claim 5) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5). In regards to Subramanian, there is a time-stamp, which functions similarly to the stage bit masks, assigned to each instruction to denote when an instruction is to be executed.

c. Wherein the buffer is further adapted to store a plurality of loop stage bit masks respectively associated with the plurality of instructions (Applicant's claims 6 and 7) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5).

d. Wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with the respective one of the loop stage bit masks associated with the certain one of the instructions (Applicant's claims 6 and 7) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5). In regards to Subramanian, there is a time-stamp, which functions similarly to the stage bit masks, assigned to each instruction to denote when an instruction is to be executed.

e. The control logic including an iteration initiation register for storing a loop iteration initiation parameter (Applicant's claims 8, 10, 12 and 14) (Subramanian column 5, lines 49-56 and Figure 5).

f. Wherein the stored plurality of instructions comprises a kernel set of loop instructions, a prologue set of loop instructions different than the kernel set of

loop instructions, and an epilogue set of loop instructions different than the kernel set of loop instructions in accordance with the kernel set of loop instructions and received loop parameters (Applicant's claims 9, 11, and 13) (Subramanian column 6, lines 4-19 and Figure 5).

13. In regards to Subramanian, the elements of the claims have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

14. Referring to claims 15 and 16, Fleck has taught a fetch unit, wherein the control logic is further operative so that the fetch unit can be shut down during execution of the stored plurality of instructions (Fleck column 1, line 50 to column 2, line 2; column 3, line 48 to column 4, line 56; Figure 1; and Figure 3).

15. Referring to claims 17 and 18, Fleck has taught a fetch unit, wherein the control logic is further operative so that the fetch unit can be shut down during execution of the loop instructions (Fleck column 1, line 50 to column 2, line 2; column 3, line 48 to column 4, line 56; Figure 1; and Figure 3). Fleck has not taught prologue, kernel, and epilogue sets of loop instructions. Subramanian has taught prologue, kernel, and epilogue sets of loop instructions (Subramanian column 6, lines 4-19 and Figure 5). In regards to Subramanian, the prologue, kernel, and

epilogue sets of loop instructions are identified as part of the necessary breakdown of a loop for modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

16. Referring to claim 26, Fleck has taught a buffer in the dispatch stage of a processor, the buffer being associated with a functional unit that executes instructions issued to it from the dispatch stage (Fleck column 1, lines 30-2; column 2, lines 26-67; Figure 1; and Figure 3). Fleck has not taught:

a. A kernel set of loop instructions;

b. A plurality of modulo schedule stage identifiers respectively associated with the kernel set of loop instructions.

17. However, Fleck has taught the buffer contains loop instructions (Fleck column 1, lines 30-2; column 2, lines 26-67; Figure 1; and Figure 3). Subramanian has taught:

a. A kernel set of loop instructions (Subramanian column 5, line 29 to column 6, line 15 and Figure 5);

b. A plurality of modulo schedule stage identifiers respectively associated with the kernel set of loop instructions (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5).

18. In regards to Subramanian, the elements of the claim have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

19. Referring to claims 27-31, Fleck has taught

a. Wherein the processor further includes control logic, the buffer being coupled to the control logic and the functional unit for causing the kernel set of loop instructions to be issued to the functional unit (Applicant's claim 27) (Fleck column 1, line 30 to column 2, line 2; column 2, lines 45-67; and Figure 1);

b. Wherein the loop instructions comprise undecoded instructions, and wherein the processor further includes a decode stage interposed between the functional unit and the buffer for decoding the instructions (Applicant's claim 30) (Fleck column 2, lines 35-55; column 3, lines 19-31; and Figure 1); and

c. Wherein the loop instructions comprise decoded instructions in the form of functional unit control signals (Applicant's claim 31) (Fleck column 2, lines 35-55; column 3, lines 19-31; and Figure 1). In regards to Fleck, instructions are functional unit control signals since they control how the functional unit operates.

20. Fleck has not taught

a. The kernel set of loop instructions and modulo schedule stage identifiers (Applicant's claim 27);

b. Wherein the modulo schedule stage identifiers comprise bit fields (Applicant's claim 28);

c. The control logic determining whether to issue a certain one of the loop instructions to the functional unit in accordance with a bit position of a set bit in the bit field corresponding to the certain loop instruction (Applicant's claim 28); and

d. Wherein the control logic is operative to cause a number of loop iterations of the kernel set of loop instructions, a prologue set of loop instructions different than the kernel set of loop instructions, and an epilogue set of loop instructions different than the kernel set of loop instructions in accordance with the stored modulo schedule stage identifiers (Applicant's claim 29).

21. Subramanian has taught:

a. The kernel set of loop instructions and modulo schedule stage identifiers (Applicant's claim 27) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5);

b. Wherein the modulo schedule stage identifiers comprise bit fields (Applicant's claim 28) (Subramanian column 3, lines 36-44);

c. The control logic determining whether to issue a certain one of the loop instructions to the functional unit in accordance with a bit position of a set bit in the bit field corresponding to the certain loop instruction (Applicant's claim 28)

(Subramanian column 2, lines 10-16; column 4, lines 16-26; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5); and

d. Wherein the control logic is operative to cause a number of loop iterations of the kernel set of loop instructions, a prologue set of loop instructions different than the kernel set of loop instructions, and an epilogue set of loop instructions different than the kernel set of loop instructions in accordance with the stored modulo schedule stage identifiers (Applicant's claim 29) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 6, lines 4-19; column 10, lines 5-9; Figure 4; and Figure 5).

22. In regards to Subramanian, the elements of the claims have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

23. Referring to claim 32, Fleck has taught a processor for executing a number of iterations of a loop comprising:

a. A plurality of functional units (Fleck column 1, line 30 to column 2, line 2; column 2, lines 26-44; and Figure 1).

b. A dispatch stage coupled to the functional units for issuing instructions to the functional units (Fleck column 1, line 30 to column 2, line 2; column 2, lines 45-67; and Figure 1).

c. A plurality of buffers respectively associated with the plurality of functional units adapted to store loop instructions (Fleck column 1, line 30 to column 2, line 2; column 2, lines 45-67; and Figure 1). In regards to Fleck, a buffer to store a loop instruction has been taught and duplicating this part is not a patentable improvement. See I*n re Harza* 274 F.2d 669, 124 USPQ 378 (CCPA 1960).

d. Control logic coupled to the plurality of buffers for causing the stored kernel set of instructions to be selectively issued to the respective functional units (Fleck column 2, lines 45-67; column 4, lines 9-56; Figure 1; and Figure 3).

24. Fleck has not taught

a. The loop including a prologue set of loop instructions, a kernel set of loop instructions and an epilogue set of loop instructions; and

b. The control logic being operative so that the functional units execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored kernel set of loop instructions.

25. Subramanian has taught

a. The loop including a prologue set of loop instructions, a kernel set of loop instructions and an epilogue set of loop instructions (Subramanian column 6, lines 4-19 and Figure 5); and

b. The control logic being operative so that the functional units execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored kernel set of loop instructions (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5).

26. In regards to Subramanian, the elements of the claim have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

27. Referring to claim 33, Fleck has taught a fetch unit, wherein the control logic is further operative so that the fetch unit can be shut down during execution of the number of iterations of the loop (Fleck column 1, line 50 to column 2, line 2; column 3, line 48 to column 4, line 56; Figure 1; and Figure 3).

28. Referring to claims 35 and 43, Fleck has taught a method for executing a number of iterations of a loop in a processor, the method comprising:

a. Storing the kernel set of loop instructions at a dispatch stage of the processor (Fleck column 1, line 30 to column 2, line 2; column 2, lines 45-67; and Figure 1); and

b. Storing loop parameters associated with the stored loop instructions in control logic of the processor (Fleck column 2, lines 45-67; column 4, lines 9-56; Figure 1; and Figure 3).

29. Fleck has not taught:

a. The loop including a prologue set of loop instructions, a kernel set of loop instructions and an epilogue set of loop instructions; and

b. Causing the stored kernel set of instructions to be selectively issued to functional units of the processor in accordance with the stored loop parameters so that the functional units of the processor execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored loop instructions.

30. Subramanian has taught:

a. The loop including a prologue set of loop instructions, a kernel set of loop instructions and an epilogue set of loop instructions (Subramanian column 6, lines 4-19 and Figure 5); and

b. Causing the stored kernel set of instructions to be selectively issued to functional units of the processor in accordance with the stored loop parameters so that the functional units of the processor execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored loop instructions (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5).

31. In regards to Subramanian, the elements of the claim have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

32. Referring to claims 36-40 and 44-48, Fleck has taught

a. A loop iteration register for storing a loop iteration parameter (Applicant's claims 36 and 44) (Fleck column 3, line 48 to column 4, line 56 and Figure 3), and

b. A loop cycles register for storing a loop cycles parameter (Applicant's claims 36 and 44) (Fleck column 3, line 48 to column 4, line 56 and Figure 3);

33. Fleck has not taught:

a. The control logic including an iteration initiation register for storing a loop initiation interval parameter (Applicant's claims 36 and 44);

b. Adjusting the value in the loop cycles register from the stored loop cycles parameter in accordance with a processor cycle (Applicant's claims 37 and 45);

c. Resetting the value in the loop cycles register in accordance with the adjusted value and the stored iteration initiation parameter (Applicant's claims 37 and 45);

d. Adjusting the value in the loop iteration register in accordance with the resetting step (Applicant's claims 38 and 46);

e. Completing the issue of loop instructions in accordance with the adjusted value in the loop iteration register (Applicant's claims 38 and 46);

f. Storing a set of modulo schedule stage identifiers respectively associated with the kernel set of loop instructions, the step of causing the stored kernel set of loop instructions to be selectively issued including the step of comparing the stored loop parameters with the modulo schedule stage identifiers (Applicant's claims 39 and 47); and

g. Wherein the step of comparing the stored loop parameters with the modulo schedule stage identifiers includes the step of indexing to a certain one of the modulo schedule stage identifiers in accordance with a current cycle in the modulo schedule stage indicated by the stored loop parameters, the step of causing the stored kernel set of loop instructions to be selectively issued further including issuing the associated loop instruction to the functional unit if the certain modulo schedule stage identifier indicates that the functional unit is active (Applicant's claims 40 and 48).

34. Subramanian has taught:

a. The control logic including an iteration initiation register for storing a loop iteration initiation parameter (Applicant's claims 36 and 44) (Subramanian column 5, lines 49-56 and Figure 5);

b. Adjusting the value in the loop cycles register from the stored loop cycles parameter in accordance with a processor cycle (Applicant's claims 37 and 45) (Subramanian column 5, line 29 to column 7, line 7; Figure 4; and Figure 5);

c. Resetting the value in the loop cycles register in accordance with the adjusted value and the stored iteration initiation parameter (Applicant's claims 37 and 45) (Subramanian column 5, line 29 to column 7, line 7; Figure 4; and Figure 5);

d. Adjusting the value in the loop iteration register in accordance with the resetting step (Applicant's claims 38 and 46) (Subramanian column 5, line 29 to column 7, line 7; Figure 4; and Figure 5);

e. Completing the issue of loop instructions in accordance with the adjusted value in the loop iteration register (Applicant's claims 38 and 46) (Subramanian column 5, line 29 to column 7, line 7; Figure 4; and Figure 5);

f. Storing a set of modulo schedule stage identifiers respectively associated with the kernel set of loop instructions, the step of causing the stored kernel set of loop instructions to be selectively issued including the step of comparing the stored loop parameters with the modulo schedule stage identifiers (Applicant's claims 39 and 47) (Subramanian column 2, lines 10-16; column 5, lines 29-45; column 10, lines 5-9; Figure 4; and Figure 5); and

g. Wherein the step of comparing the stored loop parameters with the modulo schedule stage identifiers includes the step of indexing to a certain one of the modulo schedule stage identifiers in accordance with a current cycle in the modulo schedule stage indicated by the stored loop parameters, the step of causing the stored kernel set of loop instructions to be selectively issued further including issuing the associated loop instruction to the functional unit if the certain modulo schedule stage identifier indicates that the functional unit is active

(Applicant's claims 40 and 48) (Subramanian column 5, line 29 to column 7, line 7; Figure 4; and Figure 5).

35. In regards to Subramanian, the elements of the claims have been taught by Subramanian as necessary parts of modulo scheduling. A person of ordinary skill in the art at the time the invention was made would have recognized that modulo scheduling is a form of software pipelining specifically for loops, so that successive execution iterations are overlapped (Subramanian column 2, lines 8-10), thereby increasing processor efficiency and speed by executing multiple instructions at the same time. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modulo scheduling of Subramanian in the device of Fleck to increase processor efficiency and speed.

36. Referring to claims 41 and 49, Fleck has taught shutting down the fetch unit during execution of the number of iterations of the loop (Fleck column 1, line 50 to column 2, line 2; column 3, line 48 to column 4, line 56; Figure 1; and Figure 3).

37. Claims 19-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Fleck in view of Subramanian as applied to claims 9, 11, and 13 above, and further in view of Valluri and Govindarajan's "Modulo-Variable Expansion Sensitive Scheduling" published in High Performance Computing, 1998 (herein referred to as Valluri). Fleck in view of Subramanian has not taught wherein the kernel set of loop instructions comprise modulo variable expansion (MVE) code. Valluri has taught wherein the kernel set of loop instructions comprise MVE code (Valluri section 1. Introduction, paragraphs 1-2). A person of ordinary skill in the art at the time the invention was made would have recognized that MVE is needed to handle overlapping of a single variable with a subsequent definition of itself by ensuring that different registers are used

each time (Valluri section 1. Introduction, paragraphs 1-2). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate MVE of Valluri in the device of Fleck in view of Subramanian to handle same variable overlap.

38. Claims 22-25, 34, 42, and 50 are rejected under 35 U.S.C. 103(a) as being unpatentable over Fleck in view of Subramanian as applied to claims 3, 8, 11, and 13 above, and further in view of Morrison et al., U.S. Patent Number 6,421,744 (herein referred to as Morrison). Fleck has not taught wherein the control logic is further operative to allow interrupts to be handled at the end of a current loop iteration. Morrison has taught wherein the control logic is further operative to allow interrupts to be handled at the end of a current loop iteration and to complete the number of loop iterations after the interrupt is handled (Morrison column 9, lines 18-23). In regards to Morrison, returning to complete the loop iterations is inherent to interrupts, since they are only temporary. Please see InstantWeb's Online Computing Dictionary. A person of ordinary skill in the art at the time the invention was made would have recognized that waiting until the end of a loop iteration to allow interrupts would ensure data is not lost and/or corrupted and continuing afterwards ensures the process completes and normal process has resumed. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the interrupts of Morrison in Fleck.

39. Claims 51-52 are rejected under 35 U.S.C. 103(a) as being unpatentable over Fleck et al., U.S. Patent Number 6,085,315 (herein referred to as Fleck) in view of Tran, U.S. Patent Number 5,898,849 (herein referred to as Tran). Fleck has not taught

a. Wherein the stored plurality of instructions have already been demultiplexed for issue to the functional unit and are in a form capable of execution by the functional unit (Applicant's claim 51).

b. A plurality of other functional units (Applicant's claim 52),

c. Wherein the buffer is coupled to the functional unit and adapted so that it stores instructions for execution by the functional unit but not for execution for any of the other functional units (Applicant's claim 52).

40. Tran has taught

a. Wherein the stored plurality of instructions have already been demultiplexed for issue to the functional unit and are in a form capable of execution by the functional unit (Applicant's claim 51) (Tran column 2, line 55 to column 3, line 16; column 4, line 44 to column 5, line 12; column 6. lines 31-47; and Figure 1);

b. A plurality of other functional units (Applicant's claim 52) (Tran column 2, line 55 to column 3, line 16; column 4, line 44 to column 5, line 12; column 6. lines 31-47; and Figure 1),

c. Wherein the buffer is coupled to the functional unit and adapted so that it stores instructions for execution by the functional unit but not for execution for any of the other functional units (Applicant's claim 52) (Tran column 2, line 55 to column 3, line 16; column 4, line 44 to column 5, line 12; column 6. lines 31-47; and Figure 1).

41. A person of ordinary skill in the art at the time the invention was made, and as taught by Tran, would have recognized that the reservation stations and local caches decrease cache

latency (Tran column 3, lines 2-17), thereby improving processor efficiency and speed. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the reservation stations and local cache of Tran in the device of Fleck to improve processor efficiency and speed.

Response to Arguments

42. Examiner withdraws claim objections and other informalities noted in the original claims in favor of the amended claims.

43. Applicant's arguments filed 04 April 2004 have been fully considered but they are not persuasive.

44. Applicant's argue in essence on pages 15-16

> ...Claim 1 requires that the instruction stored in the buffer must be capable of being executed by the functional unit. This is required by antecedence wherein the functional unit is defined as being "adapted to execute **an instruction**," and the claimed buffer must store "a plurality of **the instruction**." Therefore the "instruction" of the claim is an instruction in a form that is capable of being executed by the functional unit.

45. This has not been found persuasive. The claims limitations recite "A processor **comprising**: a functional unit adapted to execute an instruction **issued to it from a dispatch stage** and a buffer in the dispatch stage coupled to the functional unit adapted to store a plurality of instructions **before issue to the function unit**." The limitations suggested in the above argument are not explicitly present in the claim. Due to the open language of the claim, i.e. comprising, the claim is not limited to only a functional unit and a buffer. There may be any

number of operations and elements between when the buffer stores the instructions and when the instructions are sent to the functional unit. The claim only recites that a buffer stores instructions before they are sent to the functional stage to be executed. Fleck has taught these limitations in the passages cited in the above rejection. Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

46. Applicant's argue in essence on page 16

> First...whereas the claim requires that the control logic identifies one of the **stored** instructions to be issued to **the** identified functional unit, Fleck's alleged buffer...is positioned before the instruction demux 7...
>
> Second, Fleck's loop cache is accessed by Program Counter 2...claim 3 requires that the instructions are issued to the function unit from the buffer according to a loop iteration stage...

47. This has not been found persuasive. The claim limitations recite "...further **comprising**: control logic coupled to the buffer adapted to cause a certain one of the stored plurality of instructions to be issued to the functional unit in accordance with a loop iteration stage." The limitations suggested in the above argument are not explicitly present in the claim. Due to the open language of the claim, i.e. comprising, the claim is not limited to only the limitations in the claim. There may be additional operations and elements found in the device, as long as it is performing similar operations and has similar elements. Also, there is no limitation in the claim that excludes accessing a Program Counter from the loop iteration stage. In order to perform the loop iteration stage in Fleck, the Program Counter 2 must be accessed to know which instruction

is to be performed next. Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

48. Applicant's argue in essence on pages 17-18

> To summarize, Subramanian and its associated prior art teach how to reach a schedule from a dependence graph in compiler software, whereas the present invention teaches how to enable compact representation of the derived schedule and efficient hardware buffering and execution.

49. This has not been found persuasive. Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). The limitations applicant's suggest are explicitly present are not explicitly stated in the claims, but rather in the specification and understanding of the specification. The claim language used, i.e. "wherein the buffer is further adapted to cause the certain one of the instructions to be issued in accordance with a cycle within the loop iteration stage," does not distinguish between the compiler scheduling and compact representation, buffering and execution suggested in the arguments. The claim language, when interpreted in the broadest possible manner, merely states that a loop instruction is executed in accordance with which loop iteration the loop is in.

50. Applicant's argue in essence on pages 18-19

> In contrast, Fleck uses Program Counter 2 to access Loop Cache 3, **which are part of the fetch unit 1**. As a result, the fetch unit cannot **possible** be shut down during the execution of the stored plurality of instructions in Loop Cache 3.

51. This has not been found persuasive. First, the Examiner could not locate within Fleck where the Program Counter 2 and Loop Cache 3 were said to be part of a "fetch unit 1." Second, even if this should be labeled, the claims merely recite "wherein the control logic is further operative so that the fetch unit can be shut down during execution of the stored plurality of instructions." In the broadest interpretation, the claim limitation merely states that the instruction fetch mechanism, i.e. the mechanism that retrieves instructions from outside memory not buffer memory, is not operative, meaning that instructions from the buffer storage not outside storage are executed

52. Applicant's arguments on pages 19-23 are similar to those responded to above. Please so the responses above.

53. Applicant's argue in essence on pages 23-24

> ...In contrast Morrison merely discloses taking interrupts at the end of loop iterations. In modulo scheduled code according to the present invention, there is no natural and clean end of an iteration; all the iterations are overlapped. Accordingly, Morrison's interrupts would not meet the limitations explicitly required by the claims.

54. This has not been persuasive. Again, although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). Applicant's claims merely state "wherein the control logic is further operative to allow interrupts to be handled at the end of a current loop iteration." This limitation, when interpreted in the broadest possible manner, merely states that an interrupts can occur at the end of a loop iteration. Morrison has taught this (see

above rejection). The limitations suggested in the above argument is not explicitly stated in the claim and cannot be read upon the claim.

Conclusion

55. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

56. A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

57. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Aimee J Li whose telephone number is (571) 272-4169. The examiner can normally be reached on M-T 7:30am-5:00pm.

58. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

59. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AJL
Aimee J. Li
18 October 2004

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100